Solving the Job Shop Scheduling Problem by the Multi-Hybridization of Swarm Intelligence Techniques

Jebari Hakim[1]*, Siham Rekiek[2], Kamal Reklaoui[3]
Research Laboratory in Engineering, Innovation and Management of Industrial Systems (2ISMI), Faculty of Science and Technique of Tangier, University Abdelmalek Essaâdi, Morocco[1, 3]
The International Higher Institute of Tourism of Tangier (ISITT), Morocco[2]

***Abstract*—The industry is subject to strong competition, and customer requirements which are increasingly strong in terms of quality, cost, and deadlines. Consequently, the companies must improve their competitiveness. Scheduling is an essential tool for improving business performance. The production scheduling problem is usually an NP-hard problem, its resolution requires optimization methods dedicated to its degree of difficulty. This paper aims to develop multi-hybridization of swarm intelligence techniques to solve job shop scheduling problems. The performance of recommended techniques is evaluated by applying them to all well-known benchmark instances and comparing their results with the results of other techniques obtainable in the literature. The experiment results are concordant with other studies that have shown that the multi hybridization of swarm intelligence techniques improve the effectiveness of the method and they show how these recommended techniques affect the resolution of the job shop scheduling problem.**

Keywords—Scheduling; Job shop; Multi-hybridization; Swarm intelligence

I. INTRODUCTION

The profitability of a manufacturing company is generally achieved by minimizing production lead times, reducing stock costs, meeting deadlines, maximizing customer satisfaction and maximizing the use of machinery. These key performance indicators are dependent on effective scheduling.

Scheduling is the process of deciding how to allocate resources among the various potential job categories. The objective is to allocate resources over a period of time to complete a series of jobs. It has been the topic of a numerous publications in the area of operational research [1-4]. This is an important decision-making process in most sectors of manufacturing and services [5]. It may also be described as a decision-making process in order to optimize goals such as achieving and reducing makespan.

Scheduling problems may be modelled as allocation issues that represent a broad class of combinatorial optimization issues. In most such cases, it is very hard to come up with the optimum solution.

Job shop scheduling problem (JSSP) is a classic problem of operational search which has been regarded as a problem of combinatorial optimization difficult since the 1950s. As for the complexity of the calculations, the JSSP is an NP-hard in the strong sense of the term [6]. As a result, even in the case of very small JSSP instances, an optimum solution cannot be ensured.

In the general, job shop scheduling problem can be formulated as follows:

- There is a set of n jobs to be treated over a set of m machines.
- A job should not go on the same machine more than once.
- There are no constraints of precedence over the operations of the various jobs.
- The operations cannot be halted.
- Each machine can handle only one job at a time.
- Every job must pass through a specific sequence of predefined operations.

Every optimization issue should have an objective function that must be kept to a minimum or maximized in order to achieve a solution. In this case, the purpose of this paper is to reduce to a minimum the total time required to complete all jobs (makespan).

For job shop scheduling problems, there are numerous reference instances for the makespan minimization case:

1) FT (3) [7]: It is a collection of three instances of the combinations (n, m) ∈ {(6, 6), (10, 10), (20, 5)}.

2) LA (40) [8]: It is a collection of forty instances of which five instances of combinations (n, m) ∈ {(10, 5), (15, 5), (20, 5), (10, 10), (15, 10), (20, 10), (30, 10), (15, 15)}.

3) ABZ (5) [9]: It is a collection of five instances of the combinations (n, m) ∈ {(10, 10), (20, 15)}.

4) ORB (10) [10]: It is a collection of ten instances of the format (n, m) = (10, 10).

5) YN (4) [11]: It is a collection of four instances of the format (n, m) = (20, 20).

6) SWV (20) [12]: It is a collection of twenty instances of the combinations (n, m) ∈ {(20, 10), (20, 15), (50, 10), (50, 10)}.

*Corresponding Author

7) TA (80) [13]: It is a collection of eighty instances with ten instances for each of the formats (n, m) ∈ {(15, 15), (20, 15), (20, 20), (30, 15), (30, 30), (50, 15), (50, 20), (100, 20)}.

8) DMU (80) [14]: It is a collection of eighty instances of which ten instances for each of the combinations (n, m) ∈ {(20, 15), (20, 20), (30, 15), (30, 20), (40, 15), (40, 20), (50, 15), (50, 20)}.

Because of the NP-hard kind of job shop scheduling problems, the development of an optimum schedule is very expensive and impracticable. Hence, numerous techniques are developed for this problem.

A few contributions that have been made to solve the job shop scheduling problem are summarized in Fig. 1.

Several disadvantages have been encountered in solving the job shop scheduling problem, therefore, it became evident that concentrating on a single metaheuristic to solve the JSSP is rather limited.

A metaheuristic is rarely as powerful in diversification as it is in intensification, the solution consists in combining a technique characterized by a high exploration capability with a technique characterized by a good exploitation of the search space.

A good balance between diversification and intensification is required to obtain the optimum performance of a hybrid optimization method, hence its efficiency and success.

However, to the authors' best knowledge, very few articles can be found in the literature which investigate the multi hybridization of swarm intelligence techniques and conduct a thorough analysis.

The remainder of the paper is organized as follows: The material and method will be described in section II. The experimental findings and discussion are discussed in Section III. The conclusion is presented under Section IV.

Instances		BKS	Resolution Method	Reference	Instances		BKS	Resolution Method	Reference	Instances		BKS	Resolution Method	Reference
	FT06 (6*6)	55	BB	[18]		TA01 (15*15)	1231	EA+SS / EA+TS	[20]		DMU01 (20*15)	2563	EA+SS / EA+TS	[20]
FT	FT10 (10*10)	930	BB	[18]		TA02 (15*15)	1244	SA+TS / SA	[21]		DMU02 (20*15)	2706	EA+SS / EA+TS	[20]
	FT20 (20*5)	1165	BB	[18]		TA03 (15*15)	1218	EA+SS / EA+TS	[20]		DMU03 (20*15)	2731	EA+SS / EA+TS	[20]
	LA01 (10*5)	666	heuristics+BB / heuristics	[10]		TA04 (15*15)	1175	TS+SBP	[25]		DMU04 (20*15)	2669	EA+SS / EA+TS	[20]
	LA02 (10*5)	655	heuristics+BB / heuristics	[10]		TA05 (15*15)	1224	EA+SS / EA+TS	[20]		DMU05 (20*15)	2749	EA+SS / EA+TS	[20]
	LA03 (10*5)	597	heuristics+BB / heuristics	[10]		TA06 (15*15)	1238	EA+SS / EA+TS	[20]		DMU06 (20*20)	3244	SA	[23]
	LA04 (10*5)	590	heuristics+BB / heuristics	[10]		TA07 (15*15)	1227	EA+SS / EA+TS	[20]		DMU07 (20*20)	3046	SA	[23]
	LA05 (10*5)	593	heuristics+BB / heuristics	[10]		TA08 (15*15)	1217	EA+SS / EA+TS	[20]		DMU08 (20*20)	3188	SA	[23]
	LA06 (15*5)	926	heuristics+BB / heuristics	[10]		TA09 (15*15)	1274	EA+SS / EA+TS	[20]		DMU09 (20*20)	3092	EA+SS / EA+TS	[20]
	LA07 (15*5)	890	heuristics+BB / heuristics	[10]		TA10 (15*15)	1241	EA+SS / EA+TS	[20]		DMU10 (20*20)	2984	SA	[23]
	LA08 (15*5)	863	heuristics+BB / heuristics	[10]		TA11 (20*15)	1357	TS+CP	[16]		DMU11 (30*15)	3430	EA+TS	[24]
	LA09 (15*5)	951	heuristics+BB / heuristics	[10]		TA12 (20*15)	1367	EA+SS / EA+TS	[20]		DMU12 (30*15)	3492	TS guided by logistic regression model / TS	[26]
	LA10 (15*5)	958	heuristics+BB / heuristics	[10]		TA13 (20*15)	1342	EA+SS / EA+TS	[20]		DMU13 (30*15)	3681	EA+TS+Akers method	[19]
	LA11 (20*5)	1222	heuristics+BB / heuristics	[10]		TA14 (20*15)	1345	SA+TS / SA	[21]		DMU14 (30*15)	3394	EA+SS / EA+TS	[20]
	LA12 (20*5)	1039	heuristics+BB / heuristics	[10]		TA15 (20*15)	1339	SA	[23]		DMU15 (30*15)	3343	EA+SS / EA+TS	[20]
	LA13 (20*5)	1150	heuristics+BB / heuristics	[10]		TA16 (20*15)	1360	EA+SS / EA+TS	[20]		DMU16 (30*20)	3751	EA+TS+Akers method	[19]
	LA14 (20*5)	1292	heuristics+BB / heuristics	[10]		TA17 (20*15)	1462	EA+SS / EA+TS	[20]		DMU17 (30*20)	3814	TS guided by logistic regression model / TS	[26]
	LA15 (20*5)	1207	heuristics+BB / heuristics	[10]		TA18 (20*15)	1396	EA+SS / EA+TS	[20]		DMU18 (30*20)	3844	EA+TS+Akers method	[19]
	LA16 (10*10)	945	heuristics+BB / heuristics	[10]		TA19 (20*15)	1332	SA	[23]		DMU19 (30*20)	3765	TS guided by logistic regression model / TS	[26]
	LA17 (10*10)	784	heuristics+BB / heuristics	[10]		TA20 (20*15)	1348	SA	[23]		DMU20 (30*20)	3710	EA+TS	[24]
	LA18 (10*10)	848	heuristics+BB / heuristics	[10]		TA21 (20*20)	1642	TS+CP	[16]		DMU21 (40*15)	4380	EA+SS / EA+TS	[20]
	LA19 (10*10)	842	heuristics+BB / heuristics	[10]	TA	TA22 (20*20)	1600	EA+SS / EA+TS	[20]	DMU	DMU22 (40*15)	4725	EA+SS / EA+TS	[20]
	LA20 (10*10)	902	heuristics+BB / heuristics	[10]		TA23 (20*20)	1557	EA+SS / EA+TS	[20]		DMU23 (40*15)	4668	EA+SS / EA+TS	[20]
LA	LA21 (15*10)	1046	EA+LS+Crossover	[28]		TA24 (20*20)	1644	FDS+LNS	[27]		DMU24 (40*15)	4648	EA+SS / EA+TS	[20]
	LA22 (15*10)	927	heuristics+BB / heuristics	[10]		TA25 (20*20)	1595	INSA+NIS+TS	[22]		DMU25 (40*15)	4164	EA+SS / EA+TS	[20]
	LA23 (15*10)	1032	heuristics+BB / heuristics	[10]		TA26 (20*20)	1643	EA+TS+Akers method	[19]		DMU26 (40*20)	4647	EA+TS+Akers method	[19]
	LA24 (15*10)	935	heuristics+BB / heuristics	[10]		TA27 (20*20)	1680	EA+SS / EA+TS	[20]		DMU27 (40*20)	4848	EA+SS / EA+TS	[20]
	LA25 (15*10)	977	heuristics+BB / heuristics	[10]		TA28 (20*20)	1603	SA	[23]		DMU28 (40*20)	4692	EA+SS / EA+TS	[20]
	LA26 (20*10)	1218	heuristics+BB / heuristics	[10]		TA29 (20*20)	1625	EA+SS / EA+TS	[20]		DMU29 (40*20)	4691	EA+SS / EA+TS	[20]
	LA27 (20*10)	1235	EA+LS+Crossover	[28]		TA30 (20*20)	1584	EA+SS / EA+TS	[20]		DMU30 (40*20)	4732	EA+SS / EA+TS	[20]
	LA28 (20*10)	1216	heuristics+BB / heuristics	[10]		TA31 (30*15)	1764	EA+SS / EA+TS	[20]		DMU31 (50*15)	5640	EA+SS / EA+TS	[20]
	LA29 (20*10)	1152	EA+SS / EA+TS	[20]		TA32 (30*15)	1774	Parallel TS	[13]		DMU32 (50*15)	5927	EA+SS / EA+TS	[20]
	LA30 (20*10)	1355	heuristics+BB / heuristics	[10]		TA33 (30*15)	1791	SA	[23]		DMU33 (50*15)	5728	EA+SS / EA+TS	[20]
	LA31 (30*10)	1784	heuristics+BB / heuristics	[10]		TA34 (30*15)	1829	EA+SS / EA+TS	[20]		DMU34 (50*15)	5385	EA+SS / EA+TS	[20]
	LA32 (30*10)	1850	heuristics+BB / heuristics	[10]		TA35 (30*15)	2007	TS+SBP	[25]		DMU35 (50*15)	5635	EA+SS / EA+TS	[20]
	LA33 (30*10)	1719	heuristics+BB / heuristics	[10]		TA36 (30*15)	1819	EA+SS / EA+TS	[20]		DMU36 (50*20)	5621	EA+SS / EA+TS	[20]
	LA34 (30*10)	1721	heuristics+BB / heuristics	[10]		TA37 (30*15)	1771	EA+TS+Akers method	[19]		DMU37 (50*20)	5851	EA+SS / EA+TS	[20]
	LA35 (30*10)	1888	heuristics+BB / heuristics	[10]		TA38 (30*15)	1673	EA+SS / EA+TS	[20]		DMU38 (50*20)	5713	EA+SS / EA+TS	[20]
	LA36 (15*15)	1268	heuristics+BB / heuristics	[10]		TA39 (30*15)	1795	EA+SS / EA+TS	[20]		DMU39 (50*20)	5747	EA+SS / EA+TS	[20]
	LA37 (15*15)	1397	heuristics+BB / heuristics	[10]		TA40 (30*15)	1669	EA+TS+Akers method	[19]		DMU40 (50*20)	5577	EA+SS / EA+TS	[20]
	LA38 (15*15)	1196	SA+TS / SA	[21]		TA41 (30*20)	2005	FDS+LNS	[27]		DMU41 (20*15)	3248	EA+TS	[24]
	LA39 (15*15)	1233	heuristics+BB / heuristics	[10]		TA42 (30*20)	1937	EA+TS+Akers method	[19]		DMU42 (20*15)	3390	EA+TS	[24]
	LA40 (15*15)	1222	heuristics+BB / heuristics	[10]		TA43 (30*20)	1846	EA+TS	[24]		DMU43 (20*15)	3441	EA+TS+Akers method	[19]

ORB	ORB01 (10*10)	1059	heuristics+BB / heuristics	[10]
	ORB02 (10*10)	888	heuristics+BB / heuristics	[10]
	ORB03 (10*10)	1005	heuristics+BB / heuristics	[10]
	ORB04 (10*10)	1005	heuristics+BB / heuristics	[10]
	ORB05 (10*10)	887	heuristics+BB / heuristics	[10]
	ORB06 (10*10)	1010	Shifting Bottleneck using Guided LS / LS	[15]
	ORB07 (10*10)	397	EA+SS / EA+TS	[20]
	ORB08 (10*10)	899	Shifting Bottleneck using Guided LS / LS	[15]
	ORB09 (10*10)	934	Shifting Bottleneck using Guided LS / LS	[15]
	ORB10 (10*10)	944	Shifting Bottleneck using Guided LS / LS	[15]
SWV	SWV01 (20*10)	1407	EA+SS / EA+TS	[20]
	SWV02 (20*10)	1475	EA+SS / EA+TS	[20]
	SWV03 (20*10)	1398	EA+SS / EA+TS	[20]
	SWV04 (20*10)	1464	FDS+LNS	[27]
	SWV05 (20*10)	1424	EA+SS / EA+TS	[20]
	SWV06 (20*15)	1671	EA+TS, FDS+LNS	[24], [27]
	SWV07 (20*15)	1594	EA+TS+Akers method	[19]
	SWV08 (20*15)	1752	EA+TS, FDS+LNS	[24], [27]
	SWV09 (20*15)	1655	EA+TS, FDS+LNS	[24], [27]
	SWV10 (20*15)	1743	EA+TS+Akers method	[19]
	SWV11 (50*10)	2983	INSA+NIS+TS	[22]
	SWV12 (50*10)	2977	EA+TS	[24]
	SWV13 (50*10)	3104	EA+SS / EA+TS	[20]
	SWV14 (50*10)	2968	EA+SS / EA+TS	[20]
	SWV15 (50*10)	2885	EA+TS	[24]
	SWV16 (50*10)	2924	EA+SS / EA+TS	[20]
	SWV17 (50*10)	2794	EA+SS / EA+TS	[20]
	SWV18 (50*10)	2852	EA+SS / EA+TS	[20]
	SWV19 (50*10)	2843	EA+SS / EA+TS	[20]
	SWV20 (50*10)	2823	EA+SS / EA+TS	[20]
ABZ	ABZ5 (10*10)	1234	heuristics+BB / heuristics	[10]
	ABZ6 (10*10)	943	heuristics+BB / heuristics	[10]
	ABZ7 (20*15)	656	EA+SS / EA+TS	[20]
	ABZ8 (20*15)	665	EA+SS / EA+TS	[20]
	ABZ9 (20*15)	678	EA+LS / TS+SA	[29]
YN	YN1 (20*20)	884	EA+LS / TS+SA	[29]
	YN2 (20*20)	904	EA+TS+Akers method	[19]
	YN3 (20*20)	892	INSA+NIS+TS	[22]
	YN4 (20*20)	968	EA+SS / EA+TS	[20]

TA	TA44 (30*20)	1979	FDS+LNS	[27]
	TA45 (30*20)	2000	EA+SS / EA+TS	[20]
	TA46 (30*20)	2004	EA+TS+Akers method	[19]
	TA47 (30*20)	1889	EA+TS, FDS+LNS	[24], [27]
	TA48 (30*20)	1937	TS guided by logistic regression model / TS	[26]
	TA49 (30*20)	1961	FDS+LNS	[27]
	TA50 (30*20)	1923	EA+TS, FDS+LNS	[24], [27]
	TA51 (50*15)	2760	TS+SBP	[25]
	TA52 (50*15)	2756	TS+SBP	[25]
	TA53 (50*15)	2717	TS+SBP	[25]
	TA54 (50*15)	2839	TS+SBP	[25]
	TA55 (50*15)	2679	SA+TS / SA	[21]
	TA56 (50*15)	2781	TS+SBP	[25]
	TA57 (50*15)	2943	TS+SBP	[25]
	TA58 (50*15)	2885	TS+SBP	[25]
	TA59 (50*15)	2655	TS+SBP	[25]
	TA60 (50*15)	2723	TS+SBP	[25]
	TA61 (50*20)	2868	SA+TS / SA	[21]
	TA62 (50*20)	2869	EA+TS	[17]
	TA63 (50*20)	2755	SA+TS / SA	[21]
	TA64 (50*20)	2702	SA+TS / SA	[21]
	TA65 (50*20)	2725	SA+TS / SA	[21]
	TA66 (50*20)	2845	SA+TS / SA	[21]
	TA67 (50*20)	2825	EA+SS / EA+TS	[20]
	TA68 (50*20)	2784	SA+TS / SA	[21]
	TA69 (50*20)	3071	SA+TS / SA	[21]
	TA70 (50*20)	2995	SA+TS / SA	[21]
	TA71 (100*20)	5464	TS+SBP	[25]
	TA72 (100*20)	5181	TS+SBP	[25]
	TA73 (100*20)	5568	TS+SBP	[25]
	TA74 (100*20)	5339	TS+SBP	[25]
	TA75 (100*20)	5392	TS+SBP	[25]
	TA76 (100*20)	5342	TS+SBP	[25]
	TA77 (100*20)	5436	TS+SBP	[25]
	TA78 (100*20)	5394	TS+SBP	[25]
	TA79 (100*20)	5358	TS+SBP	[25]
	TA80 (100*20)	5183	SA+TS / SA	[21]

DMU	DMU44 (20*15)	3475	TS guided by logistic regression model / TS	[26]
	DMU45 (20*15)	3272	EA+TS+Akers method	[19]
	DMU46 (20*20)	4035	EA+TS+Akers method	[19]
	DMU47 (20*20)	3939	EA+TS+Akers method	[19]
	DMU48 (20*20)	3763	TS guided by logistic regression model / TS	[26]
	DMU49 (20*20)	3710	EA+TS	[24]
	DMU50 (20*20)	3729	EA+TS	[24]
	DMU51 (30*15)	4156	TS guided by logistic regression model / TS	[26]
	DMU52 (30*15)	4311	EA+TS	[24]
	DMU53 (30*15)	4390	TS guided by logistic regression model / TS	[26]
	DMU54 (30*15)	4362	TS guided by logistic regression model / TS	[26]
	DMU55 (30*15)	4270	TS guided by logistic regression model / TS	[26]
	DMU56 (30*20)	4941	EA+TS	[24]
	DMU57 (30*20)	4663	EA+TS	[24]
	DMU58 (30*20)	4708	EA+TS	[24]
	DMU59 (30*20)	4619	TS guided by logistic regression model / TS	[26]
	DMU60 (30*20)	4739	TS guided by logistic regression model / TS	[26]
	DMU61 (40*15)	5172	TS guided by logistic regression model / TS	[26]
	DMU62 (40*15)	5251	TS guided by logistic regression model / TS	[26]
	DMU63 (40*15)	5323	TS guided by logistic regression model / TS	[26]
	DMU64 (40*15)	5240	TS guided by logistic regression model / TS	[26]
	DMU65 (40*15)	5190	TS guided by logistic regression model / TS	[26]
	DMU66 (40*20)	5717	EA+TS	[24]
	DMU67 (40*20)	5779	TS guided by logistic regression model / TS	[26]
	DMU68 (40*20)	5765	TS guided by logistic regression model / TS	[26]
	DMU69 (40*20)	5709	EA+TS	[24]
	DMU70 (40*20)	5889	TS guided by logistic regression model / TS	[26]
	DMU71 (50*15)	6223	EA+TS	[24]
	DMU72 (50*15)	6463	TS guided by logistic regression model / TS	[26]
	DMU73 (50*15)	6153	TS guided by logistic regression model / TS	[26]
	DMU74 (50*15)	6196	TS guided by logistic regression model / TS	[26]
	DMU75 (50*15)	6189	TS guided by logistic regression model / TS	[26]
	DMU76 (50*20)	6807	TS guided by logistic regression model / TS	[26]
	DMU77 (50*20)	6792	TS guided by logistic regression model / TS	[26]
	DMU78 (50*20)	6770	EA+TS	[24]
	DMU79 (50*20)	6952	TS guided by logistic regression model / TS	[26]
	DMU80 (50*20)	6673	TS guided by logistic regression model / TS	[26]

Fig. 1. The Proposed Methods for Solving Job Shop Scheduling Problem in the Literature.

II. The Material and Method

Swarm Intelligence (SI) is considered to be one of the most important research fields that is applied by various scientists for problem-solving, computation, and solution optimization [4].

Swarm Intelligence is based on natural swarm systems and is defined as the collective problem resolution abilities of social animals. [30].

Swarm Intelligence is a direct outcome of self-organization in which the interactions of lower-level components establish an overall-level dynamic structure that can be considered intelligence [31].

Self-organization is established by four elements [30]:

1) Multiple interactions [31]: Information about food sources is shared between the employed bees and the onlooker bees on the dance floor to harvest and retrieve the food.

2) Positive feedback [31]: It is essentially a set of simple rules that assist in generating the complex structure. One example of this process is the recruitment of honeybees in a promising flower field.

3) Negative feedback [31]: Minimizes the impact of positive feedback and contributes to the creation of a counterbalance mechanism.

4) Fluctuation [31]: The scouts conduct research in the environment to randomly find the food source.

Various swarm intelligence methods have been proposed in the literature: Artificial Bee Colony (ABC) [32-35], Genetic Algorithm (GA) [36], Ant Colony Optimization (ACO) [37–40], Particle Swarm Optimization (PSO) [41], Cat Swarm (CSO) [42], Artificial Immune System [43], Bacterial Foraging [44], and Glowworm Swarm Optimization [45] and many more.

Swarm intelligence approaches are successfully used to solve many real issues and have yielded excellent results in comparison with other methods.

In this paper, the authors concentrate on two of the most popular swarm intelligence techniques, namely, Artificial Bee Colony (ABC) and Ant Colony Optimization (ACO).

A. *Fundamentals of Artificial Bee Colony Algorithm (ABC)*

The Artificial Bee Colony (ABC) is a population-based approach suggested by Karaboga and Basturk in 2007 [46] and an evolutionary algorithm based on the intelligent behavior of honeybees looking for food (nectar) [47, 48], which works by sharing information about food sources among bees in the nest.

Each position of the food source corresponds to one solution, the bees are ranked according to how they choose the food source to use. The phase of employed bee, onlooker bee and scout bee phase are the steps used in the suggested method.

The Artificial Bee Colony (ABC) algorithm has shown that it provides optimum solutions in the continuous and discreet field [1-4] [49-52] when it is confronted with noisy and multimodal optimization issues. A full overview of the use of the ABC technique is available in [47].

Analogous to other swarm techniques, the Artificial Bee Colony (ABC) method is a repetitive process.

The artificial bee colony technique (ABC) comes up with the best solution by applying four stages [53], until an end criterion is satisfied [54], these key steps of the ABC algorithm are outlined below:

1) Initialization step: It begins with a population of randomized solutions or food sources. The value of food sources determines by many factors such as their wealth, the ability to extract energy, the closeness of the nest and the amount of energy that's going to be collected from that food source.

2) Employed bee step: Every bee employed is given a source of food that it currently uses or operates. They bring together information about that particular source, their distance and direction from the nest, the cost-effectiveness of the source and they're sharing that information with some degree of probability. The number of employed honeybees will correspond to the number of food sources surrounding their beehives.

Unemployed bees wait in the nest and are seeking a source of food to explore, there are two kinds of bees unemployed:

a) Onlooker bee step: The Curious bees or Onlooker bees must look at the dance of the employed bees and then develop a source of food via information shared by the employed bees.

b) Scout bee step: Explorer or Scout bees are the ones whose source of food is abandoned and are seeking new food sources in the nesting environment. Once an adequate food source is found, scout bees become employed bees.

Register the best food source reached up to now.

From this algorithm, a colony of artificial bees (ABC) is established.

The Artificial Bee Colony (ABC) is also the same as other algorithms because it has pros and cons [55]:

1) The ABC method has a force in both local and global searches and it has been implemented with a number of optimization issues.

2) However, it has a number of parameters that need to be adjusted, randomly initialized and its local search are probabilistic.

B. Fundamentals of Ant Colony Optimization (ACO)

The Ant Colony Optimization technique (ACO) is another swarm intelligence approach based on ants' behavior looking for a food source from their nest with no visual information and using shortest pathways [56].

ACO is an optimization technique based on the population that Dorigo developed in 1992 [57] and has been successfully implemented to resolve various NP-hard combinatorial optimization issues that require to provide approximate solutions to the defined issue [58]. Algorithmically, the process of evaporating pheromones is helpful to prevent convergence towards a local optimal solution.

The steps involved in obtaining the best solution using the ant colony optimization technique are listed below [38]:

1) Building the solution space is made up of potential solutions with the help of the pheromone model.

2) These potential solutions are employed to update pheromone values in a manner that is considered to skew future sampling towards high quality solutions [59].

Ant behavior results in a self-reinforcement process: A pathway created by ants using the high level of pheromone is more followed by ants than the pathway with the low level of pheromone.

Numerous variations of ACO algorithms have been implemented in the literature [60].

C. The Proposed Methods

To enhance the original algorithms based on Swarm Intelligence, the researchers have generally hybridized them with other metaheuristics.

The Artificial Bee Colony (ABC) algorithm has the capacity to emerge from a local minimum and has a great ability to explore the global optimum which it is not immediately used, because the artificial bee colony (ABC) stocks it at every iteration.

The artificial bee colony (ABC) has been hybridized to increase its yields and effectiveness by balancing exploration and exploitation processes.

Swarm Intelligence's algorithms effectiveness is driven by two processes: exploration and exploitation [61]:

1) The exploration process: Allows the exploration of the search area in a more efficient way, and it can generate solutions that are sufficiently diverse.

2) The exploitation process: Utilizes all the information gathered from a defined problem to assist in finding new solutions that are better than the existing solutions.

An extensive exploitation and an insufficient exploration means that the system can have converge more quickly, but the likelihood of reaching the effective global optimum may be low. In addition, under-exploitation and over-exploration may result in very slow search paths to converge. Therefore, the balance between exploration and exploitation processes is crucial for achieving the optimum performance of the Swarm Intelligence method. In the literature, no non-hybridized

Swarm Intelligence algorithm is able to achieve this optimum equilibrium.

The multi-hybridization of ABC technique with ACO technique is proposed in this paper for getting a more powerful method that balances both processes, exploration and exploitation, harnesses and combines the benefits of Swarm Intelligence algorithms. The synchronous parallel hybridization [62] is applied in the proposed approach.

This approach, denoted as HABCACO involves integrating the ant colony optimization technique (ACO) into the employed bees step and/or in the onlooker bees step and/or in the scout bees step.

The HABCACO approach is described in Fig. 2.

<table>
<tr><th colspan="3">HABCACO</th></tr>
<tr><td rowspan="4">ABC</td><td>Initialization step</td><td>Uniform Random Generation</td></tr>
<tr><td>Employed bee step</td><td>Add or do not add ACO method</td></tr>
<tr><td>Onlooker bee step</td><td>Add or do not add ACO method</td></tr>
<tr><td>Scout bee step</td><td>Add or do not add ACO method</td></tr>
</table>

Fig. 2. The HABCACO Approach.

Table I shown the configuration of the HABCACO techniques.

TABLE I. THE CONFIGURATION OF THE HABCACO TECHNIQUES

	ABC		
Hybrid ABC + ACO	***Employed bee phase***	***Onlooker bee phase***	***Scout bee phase***
HABCACO1	ACO		
HABCACO2	ACO	ACO	
HABCACO3	ACO	ACO	ACO
HABCACO4		ACO	
HABCACO5		ACO	ACO
HABCACO6			ACO
HABCACO7	ACO		ACO

The suggested approaches HABCACO has ABC technique as the primary algorithm that in its flow will call ACO technique for enhancement. ACO method refines the solution generated by the steps of ABC method (employed bees step and/or onlooker bees step and/or scout bees step) and produces a better solution to be used in the process of HABCACO.

The population of candidate solutions for these new approaches is initialized using the uniform random generation between a lower bound LB and an upper bound UB [63], this means that the potential solution frequently takes the form:

$$\mathrm{CS} = \mathrm{LB} + \alpha\,(\mathrm{UB} - \mathrm{LB}), \text{where } \alpha \in [0,1] \quad (1)$$

The means to generate the candidate solution population affect an algorithm's efficiency.

Hence, the stepwise implementation of HABCACO2 and HABCACO7 are explained respectively in Fig. 3 and Fig. 4.

Initialization step:
Initialize input parameters.
Initialize the number of generations necessary for the termination criterion.
Initialize the number of solutions.
Initialize the employed bees' number (equal to solutions number)
Initialize the onlooker bees' number (equal to solutions number)
Initialize the number of scout bees
Initialize the possible solution populations by the uniform random generation.
Calculate the fitness value for each solution as follows:

$$fit_i = \frac{1}{(1+f_i)} \ \ if\ f_i \geq 0\,;\ fit_i = 1 + \left|(f_i)\right|\ if\ f_i < 0$$

Where f_i is the value of the objective function of i^{th} solutions.
Identify the best solution.

Employed bee step:
For each employed bee
Create a new solution based on the function:
$y_{ij} = x_{ij} + \emptyset_{ij}\left(x_{ij} + x_{kj}\right), k \neq i, i = \{1, 2 \ldots, SN\},$
$j = \{1, 2 \ldots, D\}, \emptyset_{ij} = Rand\,[-1, 1]$
x_{min}, x_{max} are respectively the lower and upper boundaries of the search perimeter and y_{ij} is a new realizable dimension value of the solutions which is changed from the value of its previous solutions x_{ij}.
Apply ACO (new solution).
Determine the fitness value of these best solutions found by ACO.
Calculate the most appropriate solution.

Onlooker bee step:
For each onlooker bee
Create a new solution based on the function:
$y_{ij} = x_{ij} + \emptyset_{ij}\left(x_{ij} + x_{kj}\right), k \neq i, i = \{1, 2 \ldots, SN\},$
$j = \{1, 2 \ldots, D\}, \emptyset_{ij} = Rand\,[-1, 1]$
Choose the i^{th} solution associated with the probability value (p_i):
$p_i = \frac{fit_i}{\sum_{k=1}^{SN} fit_k}$
where fit_i is the fitness value of i^{th} the solution and SN represents the number of available solutions.
Apply ACO (new solution).
Determine the fitness value of these best solutions found by ACO.
Calculate the most appropriate solution.

Scout bee step:
Create a new solution with the transmission function defined below:

$$x_i^j = x_{min}^j + rand\,[0,1]\left(x_{max}^j - x_{min}^j\right)$$

Where x_{min}^j and x_{max}^j are respectively the lower and upper boundaries of the search perimeter.
Determine the fitness value of these solutions.
Locate the best scout bee amongst the solutions produced by utilizing the fitness value.
Compare the scout bee's best solution, the employed bee's best solution and the onlooker bee's best solution by utilizing their fitness values.
Amongst these solutions, stock the best solution in the scout bee's phase and the remain solutions in the next iteration.
The procedure is repeated until the specified number of generations is attained.
The best solution is achieved with its objective value from the scout bee's phase.

Fig. 3. The Procedure of HABCACO2.

Initialization step:
Initialize input parameters.
Initialize the number of generations necessary for the termination criterion.
Initialize the number of solutions.
Initialize the employed bees' number (equal to solutions number)
Initialize the onlooker bees' number (equal to solutions number)
Initialize the number of scout bees
Initialize the possible solution populations by the uniform random generation.
Calculate the fitness value for each solution as follows:

$$fit_i = \frac{1}{(1+f_i)} \; if \; f_i \geq 0 \,; \; fit_i = 1 + |(f_i)| \; if \; f_i < 0$$

Where f_i is the value of the objective function of ith solutions.
Identify the best solution.

Employed bee step:
For each employed bee
Create a new solution based on the function:

$$y_{ij} = x_{ij} + \emptyset_{ij}\left(x_{ij} + x_{kj}\right), k \neq i\,, i = \{1,2\ \ldots, SN\},$$
$$j = \{1,2\ \ldots, D\}\,, \emptyset_{ij} = Rand\,[-1,1]$$

x_{min} , x_{max} are respectively the lower and upper boundaries of the search perimeter and y_{ij} is a new realizable dimension value of the solutions which is changed from the value of its previous solutions x_{ij}.
Apply ACO (new solution).
Determine the fitness value of these best solutions found by ACO.
Calculate the most appropriate solution.

Onlooker bee step:
For each onlooker bee
Create a new solution based on the function:

$$y_{ij} = x_{ij} + \emptyset_{ij}\left(x_{ij} + x_{kj}\right), k \neq i\,, i = \{1,2\ \ldots, SN\},$$
$$j = \{1,2\ \ldots, D\}\,, \emptyset_{ij} = Rand\,[-1,1]$$

Choose the ith solution associated with the probability value (p_i):

$$p_i = \frac{fit_i}{\sum_{k=1}^{SN} fit_k}$$

where fit_i is the fitness value of ith the solution and SN represents the number of available solutions.
Determine the fitness value of these solutions.
Calculate the most appropriate solution.

Scout bee step:
Create a new solution with the transmission function defined below:

$$x_i^j = x_{min}^j + rand\,[0,1]\left(x_{max}^j - x_{min}^j\right)$$

Where x_{min}^j and x_{max}^j are respectively the lower and upper boundaries of the search perimeter.
Apply ACO (new solution).
Determine the fitness value of these best solutions found by ACO.
Locate the best scout bee amongst the solutions produced by utilizing the fitness value.
Compare the scout bee's best solution, the employed bee's best solution and the onlooker bee's best solution by utilizing their fitness values.
Amongst these solutions, stock the best solution in the scout bee's phase and the remain solutions in the next iteration.
The procedure is repeated until the specified number of generations is attained.
The best solution is achieved with its objective value from the scout bee's phase.

Fig. 4. The Procedure of HABCACO7.

The ant colony optimization technique (ACO) is primarily made up of two phases: construction of the solution and updating of pheromone. The algorithm of ant colony optimization (ACO) technique is described in Fig. 5:

Set ACO parameters
Initialize pheromone trails
Set of potentially selected locations S = {1, 2, . . ., n}
Random selection of the initial location i
d_{ij} represents the distance between the locations i and j

Repeat
For every ant Do
Construction of solution by using pheromone trail:

Repeat
Choose new location j with probability

$$p_{ij} = \frac{(x_{ij})^{\alpha}\left({}^{1}/_{d_{ij}}\right)^{\beta}}{\sum_{k\in s}(x_{ik})^{\alpha}\left({}^{1}/_{d_{ik}}\right)^{\beta}}, \qquad \forall\, j \in S$$

$$S = S - \{j\}\,, \qquad i = j$$

Until

$$S = \emptyset$$

Update the pheromone trails:
Evaporation where the trail of pheromone automatically diminishes:
Every pheromone value is reduced by a set ratio:

$$x_{ij} = x_{ij}\,(1-\rho), \forall\, i, j \in [1, n], \rho \in\,]0,1]$$

Where ρ is the pheromone reduction rate.

The aim of evaporation is to prevent premature convergence of all ants towards the good solutions and then to promote exploration.

Reinforcement where the pheromone trail is updated in accordance with the generated solutions, the quality-based pheromone update is applied:

For every element of the best solution π^*, a positive value is added:

$$x_{\pi^*(i)} = x_{\pi^*(i)} + \Delta\,, \forall\, i \in [1, n]$$

This strategy updates the value of pheromone related to the best-found solution amongst all ants, the added values depend on the solutions chosen quality.

Until termination condition is satisfied

The best solution is achieved.

Fig. 5. The Procedure of ACO.

III. Results and Discussion

In order to justify the effectiveness of the proposed approaches HABCACO, they were simulated across a set of 250 benchmark instances from the job shop scheduling literature: FT [7], LA [8], ABZ [9], ORB [10], YN [11], SWV [12], TA [13], DMU [14], CAR [18], and compared with the best-known solution (BKS) obtained through other Techniques.

The performance of the HABCACO methods was also compared to two other advanced techniques available in the literature: HABCGA [2] and HABCPSOGA [3] to demonstrate the efficiency of HABCACO techniques in resolving job shop scheduling issues.

The HABCACO approaches have been implemented within Java and all calculation experiments were carried out on an Intel Core i7 computer with a speed of 2.5 GHz and 8 GB RAM memory under Windows 10.

All simulation and test processes were performed with the same configuration settings.

The results of the benchmark instances simulation are summarized in Table II.

The calculation results demonstrate that only the proposed HABCACO3 technique produced 100% of the best-known solution in all benchmark instances: FT (3), LA (40), ORB (10), SWV (20), ABZ (5), YN (4), TA (80), DMU (80) and CAR (8).

As shown in Table II:

1) All the suggested techniques HABCACO have given results that are 100% equal to the best-known solutions in FT (3), LA (40), ORB (10), SWV (20), ABZ (5), YN (4), and CAR (8).

2) Only the suggested techniques HABCACO3 and HABCACO7 produced 100% of the results equal to the best-known solutions in TA (80).

3) Only the suggested technique HABCACO3 produced 100% of the results equal to the best-known solutions in DMU (80).

The classification of the suggested HABCACO methods in terms of performance according to the hybridization number is illustrated in Table III.

From Table III, it can be concluded that:

1) The suggested HABCACO techniques provided the best results in comparison to the results achieved through other methods.

2) The suggested technique HABCACO hybridized in its three steps provided the best result.

3) The suggested techniques HABCACO hybridized in its two steps provided better results in comparison to the results achieved through the suggested techniques HABCACO hybridized in only one step.

4) The suggested technique HABCACO hybridized in its two steps (employed bees step and scout bees step) provided better results in comparison to the results achieved through the suggested techniques HABCACO hybridized in its two steps (onlooker bees step and scout bees step).

5) The suggested technique HABCACO hybridized in its two steps (employed bees step and scout bees step) provided better results in comparison to the results achieved through the suggested techniques HABCACO hybridized in its two steps (employed bees step and onlooker bees step).

6) The suggested technique HABCACO hybridized in its scout bees step provided better results in comparison to the results achieved through the suggested techniques HABCACO hybridized in its employed bees step or onlooker bees step.

7) The suggested technique HABCACO hybridized in its scout bees step provided better results in comparison with the results achieved through the suggested techniques HABCACO hybridized in its onlooker bees step.

The figures showed that the suggested techniques surpassed other methods with regard to the total number of benchmark instances successfully resolved and the quality of the solutions.

In order to confirm also the performance of the HABCACO technique in resolving job shop scheduling problems, it was compared to the HABCGA and HABCPSOGA methods available in the literature.

TABLE II. THE RESULTS OF BENCHMARK INSTANCES SIMULATION

HABCACO	**Benchmark Instances**																		**Total Number of Benchmark Instances**	
	FT (3)		***LA (40)***		***ORB (10)***		***SWV (20)***		***ABZ (5)***		***YN (4)***		***TA (80)***		***DMU (80)***		***CAR (8)***		***250***	
HABCACO1	3	100,00%	40	100,00%	10	100,00%	20	100,00%	5	100,00%	4	100,00%	73	91,25 %	74	92,50 %	8	100,00%	237	94,80 %
HABCACO2	3	100,00%	40	100,00%	10	100,00%	20	100,00%	5	100,00%	4	100,00%	77	96,25 %	78	97,50 %	8	100,00%	245	98,00 %
HABCACO3	3	100,00%	40	100,00%	10	100,00%	20	100,00%	5	100,00%	4	100,00%	80	100,00%	80	100,00%	8	100,00%	250	100,00%
HABCACO4	3	100,00%	40	100,00%	10	100,00%	20	100,00%	5	100,00%	4	100,00%	74	92,50 %	76	95,00 %	8	100,00%	240	96,00 %
HABCACO5	3	100,00%	40	100,00%	10	100,00%	20	100,00%	5	100,00%	4	100,00%	77	96,25 %	79	98,75 %	8	100,00%	246	98,40 %
HABCACO6	3	100,00%	40	100,00%	10	100,00%	20	100,00%	5	100,00%	4	100,00%	76	95,00 %	76	95,00 %	8	100,00%	242	96,80 %
HABCACO7	3	100,00%	40	100,00%	10	100,00%	20	100,00%	5	100,00%	4	100,00%	80	100,00%	78	97,50 %	8	100,00%	248	99,20 %

TABLE III. THE RANKING OF THE PROPOSED TECHNIQUES HABCACO

Ranking	Hybridation number	HABCACO	ABC		
			Employed bee step	*Onlooker bee step*	*Scout bee step*
7	1	HABCACO1	ACO		
4	2	HABCACO2	ACO	ACO	
1	3	HABCACO3	ACO	ACO	ACO
6	1	HABCACO4		ACO	
3	2	HABCACO5		ACO	ACO
5	1	HABCACO6			ACO
2	2	HABCACO7	ACO		ACO

As shown in Table IV:

1) The suggested technique HABCACO hybridized in its three phases with ACO method provided similar result in comparison to the results achieved through HABCGA methods hybridized in its three phases with GA method.

2) The suggested technique HABCACO hybridized in its three steps with ACO method provided better result in comparison to the results achieved through HABCGA methods hybridized in its two steps with GA method.

3) The suggested technique HABCACO hybridized in its three steps with ACO method provided better result in comparison to the results achieved through HABCGA methods hybridized in its one step with GA method.

4) The suggested technique HABCACO hybridized in its two steps with ACO method (employed bee step and scout bee step) provided better result in comparison to the results achieved through HABCGA methods hybridized in its two steps (employed bee step and scout bee step) with GA method.

5) The suggested technique HABCACO hybridized in its two steps with ACO method (onlooker bee step and scout bee step) provided better result in comparison with the results achieved through HABCGA methods hybridized in its two steps (onlooker bee step and scout bee step) with GA method.

6) The suggested technique HABCACO hybridized in it scout bee step with ACO method provided better result in comparison to the results achieved through HABCGA methods hybridized in its scout bee step with GA method.

7) The suggested technique HABCACO hybridized in it onlooker bee step with ACO method provided better result in comparison with the results achieved through HABCGA methods hybridized in its onlooker bee step with GA method.

8) The HABCGA methods hybridized in its two steps with GA method (employed bee step and onlooker bee step) provided better result in comparison to the results achieved through the suggested technique HABCACO hybridized in its two steps (employed bee step and onlooker bee step) with ACO method.

9) The HABCGA methods hybridized in its employed bee steps with GA method provided better result in comparison to the results achieved through the suggested technique HABCACO hybridized in employed bee step with ACO method.

The classification of the suggested methods HABCACO and the HABCGA methods in terms of performance according to the hybridization number and the algorithm type hybridized is tabulated in Table V.

TABLE IV. THE PERFORMANCE COMPARISON OF PROPOSED METHODS HABCACO WITH THE OTHER OPTIMIZATION ALGORITHMS HABCGA

	Hybrid ABC (ACO)						Hybrid ABC (GA)					
Hybridation number	HABCACO	ABC				Ranking	HABCGA	ABC				Ranking
		Employed bee step	*Onlooker bee step*	*Scout bee step*				*Employed bee step*	*Onlooker bee step*	*Scout bee step*		
1	HABCACO1	ACO			94,80%	7	HABCGA1	GA			95,48%	5
2	HABCACO2	ACO	ACO		98,00%	4	HABCGA2	GA	GA		99,80%	2
3	HABCACO3	ACO	ACO	ACO	100,00%	1	HABCGA3	GA	GA	GA	100,00%	1
1	HABCACO4		ACO		96,00%	6	HABCGA4		GA		94,03%	7
2	HABCACO5		ACO	ACO	98,40%	3	HABCGA5		GA	GA	96,93%	4
1	HABCACO6			ACO	96,80%	5	HABCGA6			GA	95,35%	6
2	HABCACO7	ACO		ACO	99,20%	2	HABCGA7	GA		GA	98,40%	3

TABLE V. THE RANKING OF THE PROPOSED TECHNIQUES HABCACO AND HABCGA

Ranking	Hybrid ABC (ACO) / Hybrid ABC (GA)	ABC				Hybridation number
		Employed bee step	*Onlooker bee step*	*Scout bee step*		
1	HABCACO3	ACO	ACO	ACO	100,00%	3
1	HABCGA3	GA	GA	GA	100,00%	3
2	HABCGA2	GA	GA		99,80 %	2
3	HABCACO7	ACO		ACO	99,20 %	2
4	HABCACO5		ACO	ACO	98,40 %	2
4	HABCGA7	GA		GA	98,40 %	2
5	HABCACO2	ACO	ACO		98,00 %	2
6	HABCGA5		GA	GA	96,93 %	2
7	HABCACO6			ACO	96,80 %	1
8	HABCACO4		ACO		96,00 %	1
9	HABCGA1	GA			95,48 %	1
10	HABCGA6			GA	95,35 %	1
11	HABCACO1	ACO			94,80 %	1
12	HABCGA4		GA		94,03 %	1

From Table VI it can be concluded that:

1) The suggested technique HABCACO hybridized in its three steps with ACO method provided similar result in comparison to the results achieved through HABCPSOGA methods hybridized in its three steps with GA and PSO methods.

2) The suggested technique HABCACO hybridized in its three steps with ACO method provided better result in comparison to the results achieved through HABCPSOGA methods hybridized in its two steps with GA and PSO methods.

3) The suggested technique HABCACO hybridized in its two steps with ACO method (employed bee step and scout bee step) provided equal result in comparison to the results achieved through HABCPSOGA methods hybridized in its two steps (employed bee step by GA method and scout bee step PSO method).

4) The suggested technique HABCACO hybridized in its two steps with ACO method (employed bee step and scout bee step) provided better result in comparison to the results achieved through HABCPSOGA methods hybridized in its two steps (employed bee step by PSO method and scout bee step GA method).

5) The suggested technique HABCACO hybridized in its two steps with ACO method (onlooker bee step and scout bee step) provided better result in comparison to the results achieved through HABCPSOGA methods hybridized in its two steps (onlooker bee step by GA method and scout bee step by PSO method).

6) The suggested technique HABCACO hybridized in its two steps with ACO method (onlooker bee step and scout bee step) provided better result in comparison to the results achieved through HABCPSOGA methods hybridized in its two steps (onlooker bee step by PSO method and scout bee step by GA method).

7) The suggested technique HABCACO hybridized in its two steps with ACO method (employed bee step and onlooker bee step) provided better result in comparison to the results achieved through the suggested methods HABCPSOGA hybridized in its two steps (employed bee step by GA method and onlooker bee step by PSO method).

TABLE VI. THE PERFORMANCE COMPARISON OF PROPOSED METHODS HABCACO WITH THE OTHER OPTIMIZATION ALGORITHMS HABCPSOGA

	Hybrid ABC (ACO)						**Hybrid ABC (GA // PSO)**					
Hybridation number	**HABCACO**	**ABC**				**Ranking**	**HABCPSOGA**	**ABC**				**Ranking**
		Employed bee step	***Onlooker bee step***	***Scout bee step***				***Employed bee step***	***Onlooker bee step***	***Scout bee step***		
2	HABCACO2	ACO	ACO		98,00%	4	HABCPSOGA4	GA	PSO		97,50%	7
2	HABCACO2	ACO	ACO		98,00%	4	HABCPSOGA7	PSO	GA		98,48%	3
3	HABCACO3	ACO	ACO	ACO	100,00%	1	HABCPSOGA2	GA	GA	PSO	100,00%	1
3	HABCACO3	ACO	ACO	ACO	100,00%	1	HABCPSOGA6	GA	PSO	GA	100,00%	1
3	HABCACO3	ACO	ACO	ACO	100,00%	1	HABCPSOGA9	PSO	GA	GA	100,00%	1
2	HABCACO5		ACO	ACO	98,40%	3	HABCPSOGA1		GA	PSO	97,90%	5
2	HABCACO5		ACO	ACO	98,40%	3	HABCPSOGA5		PSO	GA	97,85%	6
2	HABCACO7	ACO		ACO	99,20%	2	HABCPSOGA3	GA		PSO	99,20%	2
2	HABCACO7	ACO		ACO	99,20%	2	HABCPSOGA8	PSO		GA	98,40%	4

8) The HABCPSOGA methods hybridized in its two steps (employed bee step by GA method and onlooker bee step by PSO method) provided better result in comparison to the results achieved through the suggested technique HABCACO hybridized in its two steps with ACO method (employed bee step and onlooker bee step).

The classification of the suggested methods HABCACO and the HABCPSOGA methods in terms of performance according to the hybridization number and the algorithm type hybridized is tabulated in Table VII.

As a result, it clearly demonstrated that HABCACO has the best performance compared to other methods HABCGA and HABCPSOGA.

The proposed approaches HABCACO are robust techniques that have the potential to solve job shop scheduling problems.

One of the main limitations of this research is that it only addresses the minimization of the total time required to perform all the jobs (makespan).

TABLE VII. THE RANKING OF THE PROPOSED TECHNIQUES HABCACO AND HABCPSOGA

Ranking	**Hybrid ABC (ACO) / Hybrid ABC (GA // PSO)**	**ABC**				**Hybridation number**
		Employed bee step	***Onlooker bee step***	***Scout bee step***		
1	HABCACO3	ACO	ACO	ACO	100,00%	3
1	HABCPSOGA2	GA	GA	PSO	100,00%	3
1	HABCPSOGA6	GA	PSO	GA	100,00%	3
1	HABCPSOGA9	PSO	GA	GA	100,00%	3
2	HABCACO7	ACO		ACO	99,20%	2
2	HABCPSOGA3	GA		PSO	99,20%	2
3	HABCPSOGA7	PSO	GA		98,48%	2
4	HABCACO5		ACO	ACO	98,40%	2
4	HABCPSOGA8	PSO		GA	98,40%	2
5	HABCACO2	ACO	ACO		98,00%	2
6	HABCPSOGA1		GA	PSO	97,90%	2
7	HABCPSOGA5		PSO	GA	97,85%	2
8	HABCPSOGA4	GA	PSO		97,50%	2

IV. CONCLUSION

Because of the high level of complexity of the job shop scheduling problems, powerful and hybrid approaches are essential to address these challenging NP issues.

A robust swarm intelligence multi-hybridization technique is the key to achieving maximum efficiency in the resolution of job shop scheduling issues.

In this article, the authors develop novel multi-hybridization approaches of swarm intelligence methods called HABCACO through hybridization of artificial bee colony (ABC) algorithm and ant colony optimization (ACO) technique in various ways to provide optimal or near optimal solutions to job shop scheduling problems.

In this new approach, HABCACO adjusts the standard artificial bee colony techniques (ABC) to balance the impact of exploration and exploitation processes in algorithm performance.

Balanced exploration and exploitation capacities can improve method performance in terms of solutions quality.

The assessment of HABCACO's performance was analyzed on 250 well-known benchmark instances of the classical OR-library of job shop scheduling problems.

The overall experimental findings clearly demonstrated that the proposed new technique surpassed other compared optimization algorithms in terms of the total number of successfully resolved benchmark instances and the global optimum attainment.

Furthermore, the experimental results clearly demonstrated that the approaches are robust, effective and reliable for solving job shop scheduling problems.

More importantly, the results showed that the suggested techniques HABCACO produced the best results in comparison to other optimization methods HABCGA and HABCPSOGA available in the literature.

Therefore, the suggested approaches are solid techniques which have the potential to solve the scheduling problem and can be applied to solve complex optimization issues.

As future research, the authors intend to apply these approaches developed in this article to another type of scheduling problems and complex optimization problems.

REFERENCES

[1] H. Jebari, S. Rekiek, S. R. Elazzouzi, and H. Samadi, "Performance comparison of three hybridization categories to solve multi-objective flow shop scheduling problem: A case study from the automotive industry," International Journal of Advanced Computer Science and Applications, vol. 12, no. 4, pp. 680–689, 2021.

[2] H. Jebari, S. R. Elazzouzi, H. Samadi, and S. Rekiek, "The search of balance between diversification and intensification in artificial bee colony to solve job shop scheduling problem," Journal of Theoretical and Applied Information Technology, vol. 97, no. 2, pp. 658–673, 2019.

[3] H. Jebari, S. R. Elazzouzi, H. Samadi, and S. Rekiek, "Multi hybridization of swarm intelligence methods to solve job shop scheduling problem," Journal of Theoretical and Applied Information Technology, vol. 97, no. 16, pp. 4366–4386, 2019.

[4] H. Jebari, S. R. Elazzouzi, and H. Samadi, “The hybrid genetic algorithm for solving scheduling problems in a flexible production system,” International Journal of Computer Applications, vol. 110, no. 12, pp. 22–29, 2015.

[5] B. S. Girish and N. Jawahar, “A scheduling algorithm for flexible job shop scheduling problem,” 5th Annual IEEE Conference on Automation Science and Engineering, Bangalore, India, pp. 22–25, 2009.

[6] M. R. Garey and D. S. Johnson, “A Guide to the Theory of NP-completeness,” Computers and Intractability, Freeman, 1979.

[7] H. Fisher, and G. L. Thompson, “Industrial Scheduling,” Englewood Cliffs, NJ: Prentice-Hall, 1963.

[8] S. Lawrence, “Resource constrained project scheduling: an experimental investigation of heuristic scheduling techniques,” Pittsburgh: Graduate School of Industrial Administration, 1984.

[9] J. Adams, E. Balas, and D. Zawack, “The shifting bottleneck procedure for job-shop scheduling,” Management Science, vol. 34, no. 3, pp. 391–401, 1988.

[10] D. Applegate and W. Cook, “A computational study of the job-shop scheduling problem,” ORSA Journal on Computing, vol. 3, no. 2, pp. 149–156, 1991.

[11] T. Yamada, and R. Nakano, “A genetic algorithm applicable to large-scale job-shop problems”, Proceedings of the second International Workshop on Parallel Problem Solving from Nature (PPSN’2). Brussels, Belgium, pp. 281–290, 1992.

[12] R. H. Storer, D. Wu, and R. Vaccari, “New search spaces for sequencing problems with application to job shop scheduling,” Management Science, vol. 38, no. 10, pp. 1495–1509, 1992.

[13] E. D. Taillard, “Parallel taboo search techniques for the job shop scheduling problem,” ORSA Journal on Computing, vol. 6, no. 2, pp. 108–117, 1994.

[14] E. Demirkol, S. Mehta, and R. Uzsoy, “A computational study of shifting bottleneck procedures for shop scheduling problems,” Journal of Heuristics, vol. 3, no. 2, pp. 111–137, 1997.

[15] E. Balas and A. Vazacopoulos, “Guided Local Search with Shifting Bottleneck for Job Shop Scheduling,” Management Science, vol. 44, no. 2, pp. 262–275, 1998.

[16] J. C. Beck, T. K. Feng, and J. Watson, “Combining constraint programming and local search for job-shop scheduling,” INFORMS Journal on Computing, vol. 23, no. 1, pp. 1–14, 2011.

[17] J. P. Caldeira, “Private Communication of Result 2869 for ta62 to Éric D. Taillard, listed on Éric Taillard's Page,” 2003.

[18] J. Carlier and E. Pinson, “An algorithm for solving the job-shop problem,” Management Science, vol. 35, no. 2, pp. 164–176, 1989.

[19] J. F. Gonçalves and M. G. C. Resende, “An extended akers graphical method with a biased random-key genetic algorithm for job-shop scheduling,” International Transactions on Operational Research, vol. 21, no. 2, pp. 215–246, 2014.

[20] A. Henning, “ Praktische job-shop scheduling-probleme,” Ph.D. thesis, Friedrich-Schiller-Universität Jena, Jena, Germany, 2002.

[21] E. Nowicki and C. Smutnicki, “ A fast taboo search algorithm for the job shop problem,” Management Science, vol. 42, no. 6, pp. 783–938, 1996.

[22] E. Nowicki and C. Smutnicki, “ An advanced taboo search algorithm for the job shop problem,” Journal of Scheduling, vol. 8, no. 2, pp. 145–159, 2005.

[23] P.M. Pardalos, O. V. Shylo, and A. Vazacopoulos, “ Solving job shop scheduling problems utilizing the properties of backbone and big Valley,” Computational Optimization and Applications, vol. 47, no. 1, pp. 61–76, 2010.

[24] B. Peng, Z. Lü, and T.C.E. Cheng, “ A tabu search/path relinking algorithm to solve the job shop scheduling problem,” Computers and Operations Research, Vol. 53, pp. 154–164, 2015.

[25] F. Pezzella and E. Merelli, “ A tabu search method guided by shifting bottleneck for the job shop scheduling problem,” European Journal of Operational Research, vol. 120, no. 2, pp. 297–310, 2000.

[26] O. V. Shylo and H. Shams, “ Boosting Binary Optimization via Binary Classification: A Case Study of Job Shop Scheduling,” cs.AI/math.OC abs/1808.10813, arXiv, 2018.

[27] P. Vilím, P. Laborie, and P. Shaw, “ Failure-Directed Search for Constraint-Based Scheduling-Detailed Experimental Results,” in CPAIOR'2015, Barcelona, Spain, pp. 437–453, 2015.

[28] T. Yamada and R. Nakano, “ Genetic algorithms for job-shop scheduling problems,” In Proceedings of Modern Heuristic for Decision Support, March18-19, 1997, London, England, UK, pp. 67–81, 1997.

[29] C. Zhang, Y. Rao, and P. Li, “ An effective hybrid genetic algorithm for the job shop scheduling problem,” International Journal of Advanced Manufacturing Technology, vol. 39, pp. 965–974, 2008.

[30] E. Bonabeau, M. Dorigo, and G. Theraulaz, “ Swarm Intelligence: From Natural to Artificial Systems,” Oxford University Press, New York, NY, USA, vol. 1, 1999.

[31] E. Koc, “ The Bees Algorithm Theory, Improvements and Applications,” Ph.D Thesis, Cardiff University, Cardiff, UK, 2010.

[32] D. Karaboga, “An Idea Based On Honey Bee Swarm for Numerical Optimization,” Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, Vol. 200, 2005.

[33] V. Tereshko and A. Loengarov, “Collective decision making in honey-bee foraging dynamics,” Computing and Information Systems, vol. 9, no. 3, pp. 1, 2005.

[34] V. Tereshko, “Reaction-diffusion model of a honeybee colony’s foraging behaviour,” In International Conference on Parallel Problem Solving from Nature, Springe, Berlin, Heidelberg, pp. 807–816, September 2000.

[35] V. Tereshko and T. Lee, “How information-mapping patterns determine foraging behaviour of a honey bee colony,” Open Systems and Information Dynamics, vol. 9, no. 2, pp. 181–193, 2002.

[36] Y. Guo, X. Cao, H. Yin, and Z. Tang, “Coevolutionary optimization algorithm with dynamic sub-population size,” International Journal of Innovative Computing, Information and Control, vol. 3, no. 2, pp. 435–448, 2007.

[37] M. Dorigo and T. Stützle, “Ant Colony Optimization,” MIT Press, Cambridge, 2004. ISBN: 978-0-262-04219-2.

[38] M. Dorigo, M. Birattari, and T. Stutzle, “Ant colony optimization,” IEEE Computational Intelligence Magazine, vol. 1, no. 4, pp. 28–39, 2006.

[39] V. Maniezzo and A. Carbonaro, “Ant colony optimization: An overview,” In Essays and surveys in metaheuristics, Springer, US, pp. 469–492, 2002.

[40] T. Stützle, “Ant colony optimization,” In International Conference on Evolutionary Multi-Criterion Optimization, Springer, Berlin, Heidelberg, pp. 2–2, April 2009.

[41] J. Kennedy, “Particle swarm optimization,” In Encyclopedia of machine learning, Springer, US, pp. 760–766, 2011.

[42] S. C. Chu, P. W. Tsai, and J. S. Pan, “Cat swarm optimization,” Proc. of the 9th Pacific Rim International Conference on Artificial Intelligence, LNAI 4099, pp. 854–858, 2006.

[43] M. Bakhouya and J. Gaber, “An Immune Inspired-based Optimization Algorithm: Application to the Traveling Salesman Problem,” Advanced Modeling and Optimization, vol. 9, no. 1, pp. 105–116, 2007.

[44] K. M. Passino, “Biomimicry of Bacteria Foraging for Distributed Optimization and Control,” IEEE Control Systems Magazine, vol. 22, pp. 52–67, 2002.

[45] K. N. Krishnanand and D. Ghose, “Glowworm swarm optimization for searching higher dimensional spaces,” In Springer, Innovations in Swarm Intelligence, C. P. Lim, L. C. Jain and S. Dehuri, Eds. Heidelberg, 2009.

[46] D. Karaboga and B. Basturk, “A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm,” Journal of Global Optimization, vol. 39, no. 3, pp. 459–471, 2007.

[47] D. Karaboga, B. Gorkemli, C. Ozturk and N. Karaboga, “A comprehensive survey: artificial bee colony (abc) algorithm and applications,” Artif Intell Rev, vol. 42, no. 1, pp. 21–57, 2014.

[48] S. Ashrafinia, M. Naeem, and D. Lee, “Discrete Artificial Bee Colony for Computationally Efficient Symbol Detection in Multidevice STBC MIMO Systems,” Advances in Artificial Intelligence, 2013.

[49] T.M. PanQ-K, P. Suganthan, and T. Chua, “A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem,” Inf Sci, vol. 181, pp. 2455–2468, 2011.

[50] A. Singh, “An artificial bee colony algorithm for the leaf constrained minimum spanning tree problem,” Appl Soft Comput, vol. 9, pp. 625–631, 2009.

[51] S. Sundar and A. Singh, “A swarm intelligence approach to the quadratic multiple knapsack problem,” In ICONIP 2010, Lecture notes in computer science, Springer, Berlin, vol 6443, pp. 626–633, 2010.

[52] T.M. Pan Q-K, P. Suganthan, and AH. L. Chen, “A discrete artificial bee colony algorithm for the total flowtime minimization in permutation flow shops,” Inf Sci, vol. 181, pp. 3459–3475, 2011.

[53] Z. Beheshti and S. M. Hj. Shamsuddin, “A Review of Population-based Meta-Heuristic Algorithms,” lnt. J. Advance. Soft Comput. Appl., vol. 5, no. 1, 2013.

[54] D. Karaboga and B. Akay, “A comparative study of artificial bee colony algorithm,” Appl Math Comput, vol. 214, pp. 108–32, 2009.

[55] D. T. Pham and M. Castellani, “The bees algorithm: Modelling foraging behaviour to solve continuous optimization problems,” Sage J., vol. 223, pp. 2919–2938, 2009.

[56] M. Dorigo and M. Birattari, “Swarm intelligence,” Scholarpedia, vol. 2, no. 9, pp. 1462, 2007.

[57] M. Dorigo, “Optimization, Learning and Natural Algorithms,”, PhD thesis, Dipartimento di Elettronica, Politecnico di Milano, Italy, pp. 140, 1992.

[58] M. Dorigo, V. Maniezzo, and A. Colorni, “The ant system: optimization by a colony of cooperating agents,” IEEE Transactions on Systems, Man, and Cybernetics-Part B , vol. 26, no 1, pp. 29–41, 1996.

[59] M. Dorigo, G. D. Caro, and L. M. Gambardella, “Ant algorithms for discrete optimization,” Artificial life, vol. 5, pp. 137–172, 1999.

[60] M. Tuba and R. Jovanovic, “An Analysis of Different Variations of Ant Colony Optimization to the Minimum Weight Vertex Cover Problem,” Transactions on Information Science and Applications, vol. 6, pp. 936–945, 2009.

[61] M. Grepinšek, S. H. Liu, and M. Mernik, ” Exploration and exploitation in evolutionary algorithms: A survey,” ACM Computing Surveys, vol. 45, no. 3, pp. 1-33, 2013.

[62] E. G. Talbi, “A taxonomy of hybrid metaheuristics, ” International Journal of Heuristics, vol. 8, no. 5, pp. 541–564, 2002.

[63] X. S. Yang, “Nature-Inspired Optimization Algorithms, ” Elsevier, London, 2014.